AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

LISTING OF CLAIMS:

1. (Currently Amended) ~~a~~ A countermeasure method ~~for implementation~~ executed in an electronic component implementing a public-key cryptography algorithm ~~comprising~~ that employs exponentiation computation, with a left-to-right type exponentiation algorithm, of the type y=g^d, where g and y are elements of ~~the~~ a determined group G written in multiplicative notation, and d is a predetermined number, said countermeasure method ~~being characterized in that it includes~~ including a random draw step, at the start of or during execution of said exponentiation algorithm in deterministic or in probabilistic manner, ~~so as~~ to mask ~~the~~ an accumulator A.

2. (Currently Amended) A countermeasure method according to claim 1, ~~characterized in that~~ wherein the group G is written in additive notation.

3. (Currently Amended) A countermeasure method according to claim 1, ~~characterized in that~~ wherein the group G is the multiplicative group of a finite field written GF(q^n), where n is an integer.

4. (Currently Amended) A countermeasure method according to claim 3, ~~characterized in that~~ wherein the integer ~~is~~ n is equal to 1: n=1.

5. (Currently Amended) A countermeasure method according to claim 4, ~~characterized in that it comprises~~ comprising the following steps:

1) Determine an integer k defining the security of the masking and ~~give~~ designate d by the binary representation (d(t), d(t-1), ..., d(0))
2) Initialize the accumulator A with the integer 1

3) For i from t down to 0, do the following:

3a) Draw a random λ lying in the range 0 to k-1 and replace the accumulator A with A+λ.q (modulo k.q)

3b) Replace A with A^2 (modulo k.q)

3c) If d(i)=1, replace A with A.g (modulo k.q)

4) Return A (modulo q).

6. (Currently Amended) A countermeasure method according to claim 4, ~~characterized in that it comprises~~ comprising the following steps:

1) Determine an integer k defining the security of the masking, and ~~give~~ designate d by the binary representation (d(t), d(t-1), ..., d(0))

2) Draw a random λ lying in the range 0 to k-1 and initialize the accumulator A with the integer 1+λ.q (modulo k.q)

3) For i from t-1 down to 0, do the following:

3a) Replace A with A^2 (modulo k.q)

3b) If d(i)=1, replace A with A.g (modulo k.q)

4) Return A (modulo q).

7. (Currently Amended) A countermeasure method according to claim 2, ~~characterized in that~~ wherein the exponentiation algorithm applies to the group G of the points of an elliptic curve defined on the finite field GF(q^n).

8. (Currently Amended) A countermeasure method according to claim 7, ~~characterized in that it comprises~~ comprising the following steps:

1) Initialize the accumulator A=(A_x,A_y,A_z) with the (x,y,1) triplet and ~~give~~ designate d by the binary signed-digit representation (d(t+1), d(t), ..., d(0)) with d(t+1)=1

2) For i from t down to 0, do the following:

2a) Draw a random non-zero element λ from GF(q^n) and replace the accumulator A=(A_x,A_y,A_z) with (λ^2.A_x, λ^3.A_y, λ.A_z)

2b) Replace A=(A_x,A_y,A_z) with 2*A=(A_x,A_y,A_z) in Jacobian representation, on the elliptic curve

2c) If d(i) is non-zero, replace $A=(A_x,A_y,A_z)$ with $(A_x,A_y,A_z)+d(i)^*(x,y,1)$ in Jacobian representation on the elliptic curve

3) If $A_z=0$, return the point at infinity; otherwise return $(A_x/(A_z)^\wedge 2, A_y/(A_z)^\wedge 3)$.

9. (Currently Amended) A countermeasure method according to claim 7, ~~characterized in that it comprises~~ comprising the following steps:

1) Draw a non-zero random element λ from GF(q^n) and initialize the accumulator $A=(A_x,A_y,A_z)$ with the $(\lambda^\wedge 2.x, \lambda^\wedge 3.y, \lambda)$ triplet and ~~give~~ designate d by the binary signed-digit representation (d(t+1), d(t), ..., d(0)) with d(t+1)=1

2) For i from t down to 0, do the following:

2a) Replace $A=(A_x,A_y,A_z)$ with $2^*A=(A_x,A_y,A_z)$ in Jacobian representation, on the elliptic curve

2b) If d(i) is non-zero, replace $A=(A_x,A_y,A_z)$ with $(A_x,A_y,A_z)+d(i)^*(x,y,1)$ in Jacobian representation on the elliptic curve

3) If $A_z=0$, return the point at infinity; otherwise return $(A_x/(A_z)^\wedge 2, A_y/(A_z)^\wedge 3)$.

10. (Currently Amended) A countermeasure method according to claim 7, ~~characterized in that it comprises~~ comprising the following steps:

1) Initialize the accumulator $A=(A_x,A_y,A_z)$ with the (x,y,1) triplet and ~~give~~ designate d by the binary signed-digit representation (d(t+1), d(t), ..., d(0)) with d(t+1)=1

2) For i from t down to 0, do the following:

2a) Draw a random non-zero element λ from GF(q^n) and replace the accumulator $A=(A_x,A_y,A_z)$ with $(\lambda.A_x, \lambda.A_y, \lambda.A_z)$

2b) Replace $A=(A_x,A_y,A_z)$ with $2^*A=(A_x,A_y,A_z)$ in homogeneous representation, on the elliptic curve

2c) If d(i) is non-zero, replace $A=(A_x,A_y,A_z)$ with $(A_x,A_y,A_z)+d(i)^*(x,y,1)$ in homogeneous representation on the elliptic curve

3) If $A_z=0$, return the point at infinity; otherwise return $(A_x/A_z, A_y/A_z)$.

11. (Currently Amended) A countermeasure method according to claim 7, ~~characterized in that it comprises~~ comprising the following steps:

1) Draw a non-zero random element λ from GF(q^n) and initialize the accumulator $A=(A_x,A_y,A_z)$ with the $(\lambda.x, \lambda.y, \lambda)$ triplet and give d by the binary signed-digit representation (d(t+1), d(t), ..., d(0)) with d(t+1)=1
2) For i from t down to 0, do the following:
2a) Replace $A=(A_x,A_y,A_z)$ with $2*A=(A_x,A_y,A_z)$ in homogeneous representation, on the elliptic curve
2b) If d(i) is non-zero, replace $A=(A_x,A_y,A_z)$ with $(A_x,A_y,A_z)+d(i)*(x,y,1)$ in homogeneous representation on the elliptic curve
3) If $A_z=0$, return the point at infinity; otherwise return $(A_x/A_z, A_y/A_z)$.

12. (Currently Amended) An electronic component using the countermeasure method according to ~~any preceding claim~~ claim 1.